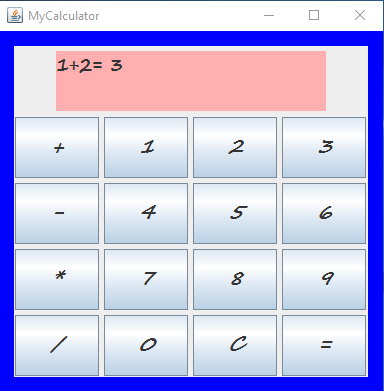
You will create a working calculator that performs the basic arithmetic operations as shown below:



You need to use arrays for buttons and the labels on the buttons. Experiment with diff fonts, colors etc. to format the calculator. I am going to be a little bit pickier with the efficiency of the code since most of this code should be a repeat from the SalesTerminal exercise.

**Challenge:**

You are to add at least one more row of functions to your calculator to increase the functionality of this simple calculator.

**Mighty Challenge:**

Rewrite your code using javascript eval function to evaluate the expression that the user enters in the text area.

**Mega Challenge:**

Open a file and write the text that is displayed in the text area to this file after each calculation.

**Challenge for a Champion:**

Display a dialogue box saying your information has been entered in the file once the calculation has been performed so that the user knows that the file operation has been successful.

**The eval function example:**

The eval() method from ScriptEngine returns the last value in the script as an Object.

**import** javax.script.ScriptEngine;

**import** javax.script.ScriptEngineManager;

**import** javax.script.ScriptException;

**public** **class** Main {

**public** **static** **void** main(String[] args) **throws** ScriptException {

ScriptEngineManager manager = **new** ScriptEngineManager();

ScriptEngine engine = manager.getEngineByName(**"JavaScript"**);

Object result = null;

result = engine.eval(**"1 + 2;"**);

System.out.println(result);

}

}

The code above generates the following result.

3

**Writing to a file Example:**

For Append mode, pass a true as second argument in FileWriter

import java.io.BufferedWriter;

import java.io.FileWriter;

import java.io.IOException;

public class FileExample1 {

public static void main(String[] args) {

String content = "This is the content to write into file\n";

// If the file doesn't exists, create and write to it

// If the file exists, append the content at the end

try (FileWriter writer = new FileWriter("app.log",true);

BufferedWriter bw = new BufferedWriter(writer)) {

bw.write(content);

bw.close();

} catch (IOException e) {

System.err.format("IOException: %s%n", e);

}

}

}

Output

app.log

This is the content to write into file

**JOptionPane Example:**

import javax.swing.\*;

public class OptionPaneExample {

JFrame f;

OptionPaneExample(){

f=new JFrame();

JOptionPane.showMessageDialog(f,"Hello, Welcome to Javatpoint.");

}

public static void main(String[] args) {

new OptionPaneExample();

}

}

Output:



Your code has to be documented and again, it has to be efficient. Follow the naming conventions and organize it so that it is easy to read.

**You need to complete all the challenges and emerge a true GUI champion!**